Coarse Grained Computations for a Micellar System.

Dmitry I. Kopelevich,[*] Athanassios Z. Panagiotopoulos,[†] and Ioannis G. Kevrekidis[‡]

Department of Chemical Engineering

Princeton University

Princeton, NJ 08544

Abstract

We establish, through coarse-grained computation, a connection between traditional, continuum numerical algorithms (initial value problems as well as fixed point algorithms) and atomistic simulations of the Larson model of micelle formation. The procedure hinges on the (expected) evolution of a few slow, coarse-grained mesoscopic observables of the MC simulation, and on (computational) time scale separation between these and the remaining "slaved", fast variables. Short bursts of appropriately initialized atomistic simulation are used to estimate the (coarse-grained, deterministic) local dynamics of the evolution of the observables. These estimates are then in turn used to accelerate the evolution to computational stationarity through traditional continuum algorithms (forward Euler integration, Newton-Raphson fixed point computation). This "equation-free" framework, bypassing the derivation of explicit, closed equations for the observables (e.g. equations of state) may provide a computational bridge between direct atomistic / stochastic simulation and the analysis of its macroscopic, system-level consequences.

I. INTRODUCTION

Textbook reaction/transport modeling is based on macroscopic equations - typically partial differential equations embodying conservation laws closed through constitutive relations. The Navier-Stokes equations as a model of laminar fluid flow provide a good ilustrative example: they describe the behavior of a very complex system (molecular collisions and interactions with the flow boundary) at a level of coarse-graining (velocity and pressure fields) which is practical for engineering design (e.g. pressure drop computations). What makes this possible is *closures*: modeling the stresses as functionals of the velocity field through Newton's law and viscosity. In engineering modeling we increasingly encounter systems whose coarse-grained, mesoscopic behavior emerges from the interaction of a number of "agents" (molecules in a fluid, cells in a tissue, individuals in a population) between themselves and with their environment. It is the coarse-grained behavior that we want to predict, design or control; yet the available models are only available at a much finer, atomistic or stochastic level. The closures that will translate these models to "practically predictive" mesoscopic level models are simply not available in closed form.

Over the last few years we have been developing an "equation free" computational methodology for extracting coarse-grained information from microscopic models; this methodology provides an alternative to direct, long-term atomistic/stochastic simulation. Our goal is to accelerate the computation of coarse-grained quantities or properties by acting directly, and as parsimoniously as possible, on the direct, full scale (kinetic Monte Carlo, kMC, molecular dynamics, MD, Brownian dynamics, BD, also quantum chemistry based simulators like such as Car-Parrinello MD, CPMD) simulator. The basic idea is to use the microscopic simulator as a computational experiment that we can initialize at will. Short bursts of appropriately initialized atomistic simulations are designed, executed, and their results processed to provide "on demand" the information that one would obtain from coarse-grained models, had these models been available in closed form. This system identification based "closure on demand" approach, provides a bridge between traditional continuum numerical analysis (integration, steady state computation, the solution of linear and nonlinear equations, optimization) and atomistic / stochastic / individual based simulators. Short, appropriately initialized dynamic simulations of the detailed model ("coarse timestepping") enable the microscopic simulator to perform tasks (like locating, quantifying the stability of, or optimizing coarse grained stationary states) that it has not been explicitly designed for. What makes it possible is the ability of a modeler to *initialize a computational experiment at will*; initializing a laboratory experiment at will is an immensely more difficult task.

This coarse-grained, equation-free computational framework was introduced in Ref. 1, and has since been used in a number of modeling contexts (from coarse Brownian dynamics of nematic liquid crystals[2] to coarse molecular dynamics for alanine dipeptide folding[3], individual-based modeling of evolving diseases[4] and lattice gas modeling of surface reactions[5,6]), see the reviews in Refs. 7,8.

The main assumption in this framework is that a separation of time scales (and concomitant space scales) prevails in the description of the system state; indeed, that we can separate the state in a subset of slow (coarse, macroscopic) variables $\mathbf{x}$ and its complement - a subset of fast variables $\mathbf{y}$. Typically, the slow variables consist of a few of the lower moments of microscopically / stochastically evolving distributions (such as concentration for chemical reactions, or density and momentum, the zeroth and first moments of a distribution of flowing molecules over velocities). Over the time scale of interest, the dynamics of the fast variables become quickly slaved to the dynamics of the slow, “master” variables. If this assumption holds (and this is a reasonable assumption if the system is believed to have reproducible coarse-grained behavior), then one does not need to derive the macroscopic equations explicitly. One can in effect solve them, and perform system-level computational modeling and design tasks with them without deriving them in closed form; model evaluations are substituted by “on demand” information from appropriately initialized microscopic simulations.

The main tool that allows the performance of numerical tasks at the macroscopic level using microscopic simulation codes is the so-called “coarse timestepper”[1]. The timestepper (which we denote by Φ_τ) is an operator which maps the state variables $\mathbf{x}(t)$ forward by time τ, i.e. $\mathbf{x}(t+\tau) = \Phi_\tau \mathbf{x}(t)$. If closed equations for the evolution of the macroscopic variables $\mathbf{x}(t)$ are available, then the timestepper is simply the solution operator for these equations for time τ. When such equations are not available, the timestepper can be estimated by the *coarse timestepper*, which we denote by Φ^c_τ. The coarse timestepper uses detailed microscopic simulations of the system, and then processes the data to estimate the time-τ map for the evolution of the macroscopic observables for the problem of interest. The coarse timestepper consists of the following basic elements (see Ref. 7):

1. A restriction operator, M, from the microscopic level description $\mathbf{y}$, to the macroscopic description, $\mathbf{x} = M\mathbf{y}$. This operator usually involves averaging over microscopic space and over realizations of the ensemble of microscopic simulations.

2. A lifting operator, μ, which constructs microscopic descriptions $\mathbf{y}$ consistent with the prescribed macroscopic description, $\mathbf{x}$. This is obviously a "one-to-many" operation. Lifting from the macroscopic to the microscopic description and then restricting (projecting) again should have no effect, i.e. $M\mu = I$.

3. Prescribe a macroscopic initial condition $\mathbf{x}(t = 0)$.

4. Transform it through lifting to an ensemble of consistent microscopic realizations, $\mathbf{y}(t = 0) = \mu\mathbf{x}(t = 0)$.

5. Evolve these realizations using the microscopic simulator for the desired short macroscopic time τ, generating the values $\mathbf{y}(\tau)$.

6. Obtain the restriction $\mathbf{x}(\tau) = M\mathbf{y}(\tau)$ and define the coarse timestepper as $\mathbf{x}(t = \tau) = \Phi^c_\tau \mathbf{x}(t = 0)$. In other words, $\Phi^c_\tau = M\Phi_\tau\mu$.

The use of such a coarse timestepper is based on (i) the separation of timescales between the micro- and the macroscopic descriptions of the system evolution, similar to that used in deriving explicit closed macroscopic equations and (ii) on the assumption of existence of a closed evolution equation *for the macroscopic variables* $\mathbf{x}(t)$. The basic premise is that the long-term dynamics of the microscopic simulation can be thought of as lying on a slow, low-dimensional manifold which is *parametrized* by the coarse variables. On this manifold, the remaining observables are slaved to (are functionals of) the chosen coarse variables. Therefore, judicious choice of the coarse variables is crucial for the correct implementation of coarse timestepping. Coarse variables typically include a few lower moments of the microscopically evolving distributions, but any appropriate order parameter set can be used as the observables of choice. Once the coarse variables and the lifting and restriction operators are constructed, the coarse timestepper procedure is easy to realize numerically. It is also easy to parallelize: each "lifted" copy for the same coarse initial condition can be executed independently on a different processor over the same wall-clock time.

Various system-level tasks can be implemented as computational superstructures around repeated calls to the timestepper. In particular, the coarse projective Euler method[7] is based on the extrapolation (projection) of the evolution of $\mathbf{x}(t)$, obtained through the application of the coarse timestepper, over some macroscopic time step Δt. A fixed point for the timestepper, i.e. a point $\mathbf{x}_0$ such that $\Phi^c_\tau(\mathbf{x}_0) = \mathbf{x}_0$, corresponds to an equilibrium configuration of a thermodynamic system; this allows one to consider using contraction mapping approaches, like the Newton-Raphson method to obtain the equilibrium states. In this work, least squares estimation is used to fit the evolution of the macroscopic observables (as opposed to the simple extrapolation in Ref. 9) and to implement fixed point computations. Other system-level tools that have been implemented using the timestepper for various systems include bifurcation and stability analysis, coarse control, coarse optimization, and the computation of coarse self-similar solutions (Ref. 7,8 and references therein.)

It is important to note that "time" and "evolution" in the above discussion do not have to be in real physical time; it could be in the wall-clock time (iteration count) of an MC simulation. In this paper we attempt to accelerate the convergence of a Monte Carlo (MC) simulation to its ultimate stationary state. We do not study how a real system dynamically approaches equilibrium; we only demand that our simulation faithfully reproduces the equilibrium, and try to accelerate the approach *of our simulation* to *its* equilibrium. It may be helpful to draw an analogy between physical evolution to a steady state, and the evolution of a Newton-Raphson iteration towards the same steady state for *deterministic* problems. The dynamics of the Newton-Raphson and the dynamics of physical evolution are different, yet they share the same fixed point. Our goal here is to accelerate (using coarse integration and coarse fixed point techniques) the convergence of *our MC computation scheme* to *its* equilibrium, which is hopefully shared with the real, dynamical problem (with "nature").

The problem we will study using these coarse-grained, equation free methods, is the spontaneous self-assembly of surfactant molecules in micelles; we attempt to accelerate the (artificial) dynamics of an MC simulation of this process.

It is well known that surfactant molecules, (amphiphiles, i.e. molecules that contain both hydrophobic and hydrophilic segments), spontaneously self-assemble into a variety of structures (see, e.g. Ref. 10). At low amphiphile concentration, above the critical micelle concentration, the surfactants assemble into micelles. We study the micellization process using grand canonical Monte Carlo (GCMC) simulations of the lattice model originally proposed by Larson[11,12]. Despite its simplicity, this model leads to predictions (in particular, phase diagrams) that agree qualitatively with experiments[12].

Recently, Panagiotopoulos and coworkers[13,14] have studied micellization using GCMC simulations of the Larson model. In order to extrapolate the results to temperature and chemical potentials different from those of actual runs, they used the histogram reweighting method[15,16]. This method allows one to compute equilibrium distributions of the number of molecules in the system N_{total} and the energy of the system E at some temperature T' and chemical potential μ' from the simulations performed at some other (nearby) T and μ. This method will not, however, extrapolate *structural* properties, such as micelle size or density; coarse computational techniques (and, in particular, Newton-Raphson based continuation) might be a promising alternative in efficiently exploring the dependence on such variables on parameters.

This paper is organized as follows: In section II we discuss the Larson model implemented in our MC simulation. In the two subsequent sections we discuss our selection of coarse variables (Section III) and our lifting procedure (the construction of microscopic states consistent with coarse observables) (Section IV). We then discuss and illustrate coarse projective integration (Section V) and the coarse Newton method (Section VI). In Section VII we discuss a kMC model fitted from our MC simulations, and apply the same coarse-grained computational methods to it. We summarize our results and conclude with a discussion in the final section (VIII).

II. THE MODEL

In the Larson model[11,12] studied here, three-dimensional space is discretized into a cubic lattice. An amphiphile molecule is modeled as a chain of beads and a solvent molecule is modeled by a single bead. The beads occupy sites on the lattice and the connected beads of an amphiphile molecule are located on the nearest-neighbor sites located along vectors $(0, 0, 1)$, $(0, 1, 1)$, $(1, 1, 1)$ and their reflections along the principal axis, resulting in a coordination number of 26. There are two types of beads: hydrophobic tail (T) and hydrophilic head (H). The solvent beads are assumed to be identical to the head beads.

Hydrophobic interaction is modeled by attractive interaction between the tail beads. Each bead interacts only with its 26 nearest neighbors and the total energy of the system is the sum of pairwise interactions between beads. The tail-tail interaction energy ϵ_{TT} is -2 and the tail-head and head-head interaction energies ϵ_{TH} and ϵ_{HH} are zero, following Ref. 14. It is furthermore assumed that all sites that are not occupied by the amphiphile beads are occupied by the solvent. This latter assumption implies that there is no need to explicitly consider solvent in the Monte Carlo (MC) moves.

We perform grand canonical Monte Carlo (GCMC) simulations of this model and use the following mix of MC moves: 50% amphiphile transfers (i.e. addition or removal), 49.5% amphiphile partial regrowth moves, and 0.5% cluster moves. We perform simulations of a linear amphiphile chain H_4T_4 (which consists of 4 head beads and 4 tail beads) in a cubic box with the side length of 40 sites, assuming periodic boundary conditions. This simulation box size is sufficiently large to prevent spurious effects of periodicity: a typical diameter of a micelle observed in the simulations reported here is significantly smaller than half the size of the box side.

A snapshot of a simulation is shown in Fig. 1. In this example, we observe that the surfactant molecules have formed three micelles. In addition, there is a significant fraction of smaller clusters. Our working definition of a cluster is an aggregate of amphiphile molecules such that each molecule in a cluster has at least one tail bead which occupies a neighboring site with a tail bead from another amphiphile of the cluster. In other words, each cluster molecule interacts through hydrophobic attraction with at least one other cluster molecule. The cluster size is defined as the number of amphiphiles in this cluster. Note that an isolated amphiphile molecule can be viewed as a cluster of size 1.

An example of the cluster size distribution obtained from the GCMC simulations is shown in Fig. 2a and is typical for a micellar system. There are two peaks in the distribution – one peak corresponding to the small clusters and the other peak corresponding to micelles. Note that there is an almost vanishing probability to observe a cluster of an intermediate size. This observation is very important and allows us to define a set of coarse variables (or "coarse observables").

III. COARSE VARIABLES

Choosing an appropriate set of coarse variables ("observables") is an important step in the implementation of equation-free computation. This choice is system-specific and should be guided by physical intuition about the system, or by data analysis techniques (e.g. Ref. 17,18). One of the requirements for the coarse variables is that they provide sufficient information about the system so that the "lifting" operation (micro from macro) can be successfully performed. In earlier work on coarse computation[2,5,6], coarse variables were chosen to be moments of an evolving distribution. Lifting then consisted of generating random realizations of a system configuration such that the average and, possibly, the variance of this quantity agreed with the prescribed values. More sophisticated lifting techniques (including short, constrained simulations in the spirit of algorithms like SHAKE in molecular dynamics[19]) are also being developed and tried. Simple lifting approaches based on only a couple of moments may not be satisfactory for more complex systems such as the one studied in the current work. In addition to the averages, we also need to preserve more of the structure of the system during the transfer of information between micro and macro (lifting and restricting). More sophisticated approaches, which take into account the spatial inhomogeneity of a system, have been presented by Gear *et al.*[20]. In this so called micro-Galerkin method, an expansion of the spatial distribution of the quantity of interest in a set of global basis functions was used; these can be traditional polynomial basis functions[21] or possibly empirical eigenfunctions obtained by principal component data analysis from a microscopic simulation.

Principal components of the raw data would give us some averaged structural information about a micelle, e.g. a density profile. It is not, however, trivial to reconstruct a micelle from its density profile. We believe that nonlinear data analysis techniques[17,18] hold the key to systematic, non-intuitive choices of appropriate coarse observables.

In the case of our micellar system we kept a database of cluster structures, and used as coarse observables a number of features of the distribution of such structures; we will describe these observables below. In the lifting procedure, we place cluster structures from the database directly into the simulation box according to certain prescribed distribution features. We envision that for more complex self-assembled systems, such as bilayer or vesicles, one can combine the empirical eigenfunction method with maintaining a structure database.

Details on computing the structure database will be provided in section V. In addition to the database, we need to specify distributions of several quantities. Describing these distributions through a finite number of macroscopic observables (coarse variables) will be the basis of our coarse timestepper. To choose a reasonable set of observables, capable of parametrizing the evolution of our simulations, we examine representative simulation runs. Clearly, one needs to know the distribution of clusters. In this work, we assume that the only quantity needed to characterize a cluster is the number of amphiphiles. That is, all other physical properties of the cluster (such as its radii of gyration and its energy) are quickly slaved to the cluster size. If we monitor clusters of a certain size, we observe that over very short periods of simulation time, their physical attributes will "approach a slow manifold" – that is, their statistics will very quickly become functions of the size. The analysis in the companion paper[22] shows that this is a reasonable assumption for the system considered here. Hence, in order to characterize the clusters contained in the system, we only specify their size distribution shown in Fig. 2a.

In addition to the distribution of cluster sizes, we need to specify the concentration of amphiphiles in the system. Since we are performing simulations in the grand canonical ensemble, in an equilibrium state, this concentration will fluctuate around some average value. In principle, for a system of fixed volume, to specify concentration, one can specify the total number N_{total} of amphiphile molecules in the system. However, we found it more convenient to "split" this N_{total} into the "small cluster" and the "micellar components". This allows us to monitor the density of micelles directly from our coarse variables and to take advantage of the separation of time scales between the micelles and the small clusters, which is discussed below. We furthermore assume that there is no correlation between sizes of different clusters in the system. This assumption is reasonable because there are no long-range interactions in the system and the solution non-idealities are only due to excluded volume interactions. The systems considered here are dilute and hence the non-idealities (correlations) are expected to be small. We will show it later that this assumption indeed holds.

To summarize, we suggest that the system can be macroscopically successfully characterized by distributions of the following three quantities (see Fig. 2): (i) cluster size, (ii) number N_{micelle} of micelles in the system, and (iii) number N_{small} of molecules contained in small clusters.

A typical distribution of cluster sizes is shown in Fig. 2a. Clearly, the clusters can be divided into two classes – small clusters and micelles. Small clusters are unstable aggregates of a small number of molecules, usually less than 10. They are being formed and destroyed very quickly during a simulation. For definiteness, we classify a cluster as a micelle if its size exceeds 20 molecules and as a small cluster otherwise. Since there is essentially zero concentration of clusters of size between 20 and 40, a precise location of the border between micelles and small clusters is unimportant.

The cluster size distribution is further approximated by a Gaussian distribution (dashed line in Fig. 2a) and the small cluster size distribution is approximated by a Poisson distribution (dotted line in Fig. 2a),

$$P(k) = \frac{\lambda^{k-1}}{(k-1)!} e^{-\lambda}, \tag{1}$$

where $k = 1, 2, 3, \ldots$ is the cluster size and $(\lambda + 1)$ is the average size of a small cluster. The size distribution of small clusters shows good agreement with the Poisson distribution because the probability of formation of a small cluster consisting of k molecules (in an ideal solution) is proportional to λ^k.

Hence, the cluster size distribution is described by three parameters: average ν and standard deviation σ of the micelle size and the parameter λ of the Poisson distribution of the small cluster sizes. One more parameter is needed to completely specify the cluster size distribution – namely, the ratio between numbers of molecules contained in the small clusters and in the micelles. This information is contained in the distributions of N_{small} and N_{micelle} and the section IV describes the details of reconstruction of the system from these parameters.

Typical examples of distributions of N_{small} and N_{micelle} are shown in Fig. 2b and 2c, respectively. The distribution of number N_{small} of molecules contained in small clusters is approximated by a Gaussian distribution

$$P(N_{\text{small}}) = P_{\text{Gauss}}(N_{\text{small}}; \nu_s, \sigma_s). \tag{2}$$

Here, ν_s and σ_s are the mean and the standard deviation of N_{small} and

$$P_{\text{Gauss}}(x; \nu, \sigma) = \frac{1}{\sqrt{2\pi\sigma}} \ \exp(-(x-\nu)^2/2\sigma^2) \tag{3}$$

is the Gaussian distribution with the mean ν and the standard deviation σ. As seen in Fig. 2b, the Gaussian distribution provides a good approximation to $P(N_{\text{small}})$ and hence in our coarse simulation, the distribution of N_{small} is completely specified by its first two moments, ν_m and σ_s.

The distribution of the number of micelles N_{micelle} is approximated by a Gaussian truncated at $N_{\text{micelle}} = 0$, i.e.

$$P(N_{\text{micelle}}) = CP_{\text{Gauss}}(N_{\text{micelle}}; \hat{\nu}_m, \hat{\sigma}_m), \text{ if } N_{\text{micelle}} \geq 0, \tag{4}$$

$$P(N_{\text{micelle}}) = 0, \text{ if } N_{\text{micelle}} < 0. \tag{5}$$

Here, C is a normalization constant. This truncation is important, because there is a non-vanishing probability to have zero micelles in the simulation box (see Fig. 2c). If we were to fit an untruncated Gaussian to such a distribution, we would end up with a non-vanishing probability to have a negative number of micelles in the simulation box. Note that, unlike the standard Gaussian, the parameters $\hat{\nu}_m$ and $\hat{\sigma}_m$ of P_{Gauss} in Eq. (4) do not coincide with the mean ν_m and and the standard deviation σ_m of N_{micelle}. It is therefore necessary to obtain these parameters from the requirement that the mean and the standard deviation of the truncated distribution coincide with the mean ν_m and the standard deviation σ_m of N_{micelle}. This is done via solution of a system of two nonlinear equations:

$$\nu_m = \sum P(N_{\text{micelle}})N_{\text{micelle}}, \qquad \sigma_m^2 = \sum P(N_{\text{micelle}})N_{\text{micelle}}^2 - \nu_m^2. \tag{6}$$

Note that these equations have a solution for only a subset of values of ν_m and σ_m. Fig. 2c shows that the truncated Gaussian provides a good approximation for $P(N_{\text{micelle}})$. Although we modified the standard Gaussian distribution to fit $P(N_{\text{micelle}})$, the distribution of the number of micelles is still specified by its first two moments ν_m and σ_m.

To summarize, we chose to model the system by 7 coarse variables (observables): 3 for small clusters (λ, ν_s, σ_s) and 4 for micelles (ν, σ, ν_m, σ_m).

Timescales (in MC iteration "time") for the evolution of these coarse variables are illustrated in Figures 3 and 4 which show evolutions of the coarse variables when the system parameters (temperature T and chemical potential μ) are switched from $k_BT = 7.5$, $\mu = -46.20$ to $k_BT = 7.0$, $\mu = -47.40$. The small cluster variables are shown in Fig. 3 and are extremely fast - they approach the slow manifold on the timescale of less than 0.1 million steps. The micelle size distribution is also relatively fast – as Figs. 4a and 4b show, equilibration time of the first two moments of the micelle size distribution is on the order of 10 million steps. The slowest dynamics is that of the micelle number distribution – it equilibrates on the order of 1000 million steps (see Figs. 4c and 4d). The observed timescales suggest that the coarse projective integration should be performed for only two variables – ν_m and σ_m – average and standard deviation of number of micelles.

The described choice of coarse variables assumes that there is no correlation between the number of micelles N_{micelle} in the system and the micelle size. In order to check this assumption, we computed the correlation coefficient between N_{micelle} and the average size of a micelle at different values of temperature T and the chemical potential μ. Results of this calculation are shown in Figure 5. For relatively dense systems, the correlation coefficient is non-vanishing and we might need to take it into account when working at those conditions. However, for low surfactant density (thick line in Fig. 5), the correlation is negligible. The coarse integration results below are reported for this low density.

IV. THE LIFTING PROCEDURE

In this section, we describe the lifting procedure, i.e. how realizations of the detailed system are reconstructed from the 7 coarse variables. Since there is no long-range interaction between the clusters in the system and we consider sufficiently dilute systems, the lifting procedure consists of the following two stages: first, we generate a sequence of sizes of clusters to be placed into the system. Second, we place these clusters into the simulation box. In order to improve efficiency of the second stage of lifting, we sort the list of cluster sizes generated in the first stage in descending order. This is done because it is easier to place a larger cluster into an emptier system and it does not introduce any bias since correlations between different clusters in the system are negligible. For each cluster size from the list, we randomly pick a cluster from a database and place it into a random location in the simulation box. The only requirement in this procedure is that the clusters do not overlap or touch each other. In principle, this procedure can be generalized to the case of denser systems or systems with long-range interactions, where the correlations between clusters become important (see also Ref. 6, where pair probabilities between adsorbate atoms must be appropriately initialized).

Let us now describe the generation of the cluster size list in more detail. This procedure is split into two parts. First, the micelle sizes are generated: the number of micelles N_{micelle} in the system is sampled from the truncated Gaussian distribution Eq. (3) with mean ν_m and standard deviation σ_m. Then, for $j = 1, \ldots, N_{\mathrm{micelle}}$, the j-th micelle size is sampled from the Gaussian distribution with parameters ν and σ.

The number N_{small} of molecules contained in small clusters is sampled from the Gaussian distribution with parameters σ_s and ν_s. Then the cluster sizes N_j, $j = 1, 2, \ldots$ are sampled from the Poisson distribution Eq. (1) with parameter λ. We stop after we have generated M clusters such that

$$\sum_{j=1}^{M-1} N_j \leq N_{\text{small}} < \sum_{j=1}^{M} N_j. \tag{7}$$

We then keep all of these clusters if the sum on the right hand side of Eq (7) is closer to N_{small} than the sum on the left hand side. Otherwise, we keep only the first $M - 1$ clusters generated. Although this procedure does not always produce exactly N_{small} molecules in each realization, it does so on average. Moreover, as we have shown, the small cluster variables are slaved to the micellar variables. Therefore, small errors in small clusters lifting will be quickly healed in the microscopic simulation. If necessary, the short "healing" step can be implemented in a constrained fashion (evolving the MC simulation with a hard parabolic potential around the target values of the slow observables, as in umbrella sampling[23]).

V. COARSE PROJECTIVE INTEGRATION

In this section we describe coarse integration of the micellar system using the coarse projective Euler method. This method is briefly outlined in the introduction and is discussed in more detail by Gear and Kevrekidis[9].

Other coarse integration methods are also proposed in Ref. 9,24 and their linear stability analysis is performed.

Before we start the coarse integration, we perform the microscopic (GCMC) simulation and let all the fast modes equilibrate. Fast modes here refer to the small cluster variables and the first two moments of the distribution of micelle sizes. As shown in section III, these modes are much faster than the mean ν_m and the variance σ_m of the number of micelles in the simulation box. After the "fast mode equilibration" is complete, we let the simulations run for a little longer in order to collect enough samples of clusters for the database. We run this "database production" simulation for an additional 5 million steps and use 500 copies of the system. The database is updated after each 0.1 million steps. This step size is sufficiently large for the micelle structures to be significantly modified and hence we populate the database with statistically different clusters. A more detailed analysis of the rate of change of a cluster structure will be presented in the companion paper[22].

When the initial preparations are complete, we perform the coarse projection. The lifting procedure was described in the previous section. After the lifting, we let the system "heal" for $t_{\text{healing}} = 0.2$ million steps. This time is more than sufficient to eliminate any discrepancies in the (fast) small cluster variables λ, ν_s, and σ_s that might be introduced during the lifting – see the earlier discussion of timescales. Since we pick the micelle structures directly from the "equilibrium" database, we expect that no healing is needed for the micelle size variables ν and σ. As mentioned above, the initial "healing" preparatory step can also be implemented in a constrained fashion as in umbrella sampling[23].

After the healing is complete, we perform the (unconstrained) simulation for additional $t_{\text{generation}} = 9.8$ million steps. We then fit a straight line through the computed $\nu_m(t)$ and $\sigma_m(t)$ and extrapolate these quantities for the macroscopic step $\Delta t = 50$ million steps,

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \mathbf{F}(\mathbf{x}(t)). \tag{8}$$

Here, $\mathbf{x}(t) = (\nu_m(t), \sigma_m(t))$ are the slow coarse variables and $\mathbf{F}(\mathbf{x}(t))$ is their slope obtained from the least squares fit to the results of the production run of length $t_{\text{generation}} = 9.8$ million steps. The fast coarse variables are set to the values from the last step of the previous microscopic simulation run.

Results of the coarse integration are shown in Fig. 6. We observe good agreement -at the level of the macroscopic observables- with the control full scale GCMC run, also shown in this figure. These simulations are performed for a system switching from $k_BT = 7.5$, $\mu = -46.20$ to $k_BT = 7.0$, $\mu = -47.40$. We used different numbers of copies of the system in the coarse integration and the control run: in the coarse integration, we used 2000 copies and in the control run we used 500 copies. The number of copies had to be increased for the coarse integration in order to reduce statistical error in the extrapolations: the projection step "magnifies" the noise and one has to have precise data in order to obtain a reasonable accuracy during extrapolation. Hence, there is a trade off between the size of the projection step and number of copies of the system needed for an accurate extrapolation.

In our particular case, the savings of CPU time in the coarse integration (as compared to the control run) are about 50%. The efficiency of the coarse Euler scheme is the subject of further investigation (see Section VII) and possible improvements to the method are discussed in Section VIII.

VI. COARSE NEWTON METHOD

In addition to coarse projective integration, one can perform other system-level tasks, such as fixed point location and stability analysis, using the information obtained from the short-scale simulations. In this section, we describe an application of a Newton-like method to the micellar system. The approach is based on the observation that the equilibrium configuration of the system corresponds to a solution of a nonlinear system of equations

$$\mathbf{F}(\mathbf{x}) = 0, \tag{9}$$

where F_i is the slope of the evolution of the i-th coarse variable $X_i(t)$. The slope of $\mathbf{F}$ is estimated by fitting a straight line to results of the short-scale microscopic simulations. Hence, one can use the Newton method (or some other method of solution of nonlinear equations) in order to obtain the equilibrium configuration of the system.

In a Newton algorithm for a deterministic problem one evaluates the residual at the current guess, the Jacobian of the equations at the current guess, and solves a linear set of equations to provide the next guess, at which the procedure is repeated. The construction of variants of the Newton method appropriate for fixed point computations in noisy environment is the subject of ongoing research (e.g. the stochastic approximation algorithms[25,26]).

In this paper we try to estimate the deterministic component of the noisy simulation, reducing the variance through averaging a number of copies, and perform Newton-Raphson on the deterministic part. The derivatives required in the Newton step will be estimated using finite differences (i.e. by initializing *macroscopically nearby* initial conditions, and observing the difference in the evolution of their coarse variables).

One important practical twist in our implementation of the Newton method for a noisy/stochastic system (compared to deterministic noise-free system of equations) is that our lifting procedure does not create microscopic states *precisely* consistent with the macroscopic observables; rather, it creates microscopic states consistent with slightly *nearby* observables. In the Newton context, the function $\mathbf{F}$ is computed not for the specified observable values $\mathbf{x}$ of the coarse variables but for slightly different values. This happens due to the stochastic nature of our lifting procedure, which involves sampling of the random variables according to a prescribed distribution. Of course, if the sample size is infinitely large, then the moments of the sampled realizations of the system will coincide with the prescribed ones. However, the finite sample size introduces some statistical errors in the generated moments. It is also possible, *after* lifting and *before* the Newton procedure, to apply an additional preparatory step, in which the microscopic state is adjusted to correspond to the macroscopic observables exactly (either through constrained evolution or some sort of simulated annealing). Such reasonable modifications of the procedure are the subject of current research; we are proceeding with our current lifting operator.

To be more specific, consider the lifting procedure for micellar systems, which involves sampling of the number of micelles N_{micelle} from the first two moments of the truncated Gaussian distribution. I.e. in this case the coarse variables are $\mathbf{x} = (\nu_m, \sigma_m)$. We observe that for the values of ν_m and σ_m typical for our simulations, one needs at least 10^4 realizations in order to sample an accurate mean ν_m and standard deviation σ_m. This large value of realizations is impractical for the micellar simulations and we have used 2000 copies in our implementation of the Newton method.

The implication of this statistical error with our current lifting is that it is necessary to modify the standard finite-difference calculations of the Jacobian. In what follows, we first describe the necessary changes for a one-dimensional function and then generalize them to a system of arbitrary dimensionality. Consider the standard forward difference estimation of a derivative,

$$\frac{dF(x)}{dx} = \frac{F(x+\Delta x) - F(x)}{\Delta x} + O(\Delta x) \tag{10}$$

However, in the lifting, instead of x and $x + \Delta x$, we generate $L(x)$ and $L(x + \Delta x)$, where $L(x)$ denotes the value of a coarse variable which is generated during the lifting step from the prescribed value x. We expect that, as the number of copies of the system approaches infinity, $L(x)$ approaches x and therefore, for a finite but large number of copies, $L(x)$ is not very different from x. Hence, instead of $F(x)$ and $F(x+\Delta x)$, we compute $F(L(x))$ and $F(L(x + \Delta x))$ and thus the correct forward difference estimate for the derivative is

$$\frac{dF(x)}{dx} = \frac{F(L(x+\Delta x)) - F(L(x))}{L(x+\Delta x) - L(x)} + O(\Delta x). \tag{11}$$

We found that this correction to the standard forward difference formula (10) is significant in the implementation of the coarse Newton method for the micellar system and that taking it into account have improved the convergence of the method.

The correction to the forward-difference scheme described above is the simplest estimate for the derivative in our noisy system. Without more precise lifting, we do not expect standard higher order schemes to yield significant improvement in the Newton convergence. There is significant recent interest in development of higher order difference schemes for derivative estimation in noisy systems. In particular, Gear[27] has proposed to use the least-squares fit and Drews *et al.*[28] have developed a central difference scheme with the weights chosen in order to reduce the variance of the derivative estimate. Their estimate is still $O(\Delta x)$ but the variance of $F'(x)$ is reduced. The work on testing and developing such schemes is currently in progress.

In this paper, we use the corrected forward-difference scheme Eq. (11). Let us briefly discuss the generalization of this formula to multiple dimensions while keeping in mind the standard forward-difference scheme,

$$\frac{\partial F_i(\mathbf{x})}{\partial x_j} = \frac{F_i(\mathbf{x}+\Delta x_j \mathbf{e_j}) - F_i(\mathbf{x})}{\Delta x_j} + O(\|\mathbf{x}\|), \quad i, j = 1, \ldots, N. \tag{12}$$

Here, N is the dimensionality of the (coarse) system, $\mathbf{e_j}$ is a unit vector pointing in the j-th direction and Δx_j is an increment in x_j. Due to the uncertainties in the lifting procedure, we actually compute the function $\mathbf{F}(\mathbf{x})$ at points $\mathbf{x_0} \equiv L(\mathbf{x})$ and $\mathbf{x_j} \equiv L(\mathbf{x} + \Delta x_j \mathbf{e_j})$. From the Taylor expansion of the function $F_i(\mathbf{x})$,

$$F_i(\mathbf{x_j}) = F_i(\mathbf{x_0}) + \nabla F_i(\mathbf{x_0}) \cdot (\mathbf{x_j} - \mathbf{x_0}) + O(\|(\mathbf{x_j} - \mathbf{x_0}\|), \tag{13}$$

it follows that

$$\nabla F_i(\mathbf{x_0}) = A^{-1}\mathbf{g}, \tag{14}$$

where $O(\|(\mathbf{x_j} - \mathbf{x_0}\|)$ terms have been neglected and $g_j \equiv F_i(\mathbf{x_j}) - F_i(\mathbf{x_0})$ and $A_{kj} \equiv x_{kj} - x_{0j}$, $j, k = 1, \ldots, N$ (x_{kj} denotes the k-th component of the vector $\mathbf{x_j}$). This formula, Eq. (14), is similar to the forward difference simplex gradient described, e.g., in Ref. 29 in the context of optimization of noisy functions. It is possible to use this formula with less or more than $N + 1$ evaluations of function $\mathbf{F}(\mathbf{x})$. In the first case, the solution of Eq. (14) will be given by a pseudoinverse of the matrix A and the computation will be somewhat similar to the simultaneous perturbation stochastic approximation method[30].

In the second case, the estimate of the Jacobian will be given by the least squares fit[27].

In the current work, we limit the estimation of the derivative to the direct use of Eq (14) and hence we perform $(N + 1)$ evaluations of the left-hand-side per iteration of the Newton method. We choose the increment Δx_j of x_j in the calculation of the derivative to be an integer multiple of the statistical error x_{err} of x_j. This choice assures that the increments of different coarse variables are not too small (i.e. not smaller than the noise). The stopping criterion for the iterations is

$$F_i(x) \leq F_{i,err}(x),\ i = 1, \ldots, N, \tag{15}$$

where $F_{i,err}$ is the statistical error estimate for the function F_i.

We perform coarse Newton simulations for the two slowest coarse variables, $\mathbf{x}(t) = (\nu_m, \sigma_m)$. The function $\mathbf{F}(\mathbf{x})$, i.e. the slope of $\mathbf{x}$(t), is computed by fitting a straight line to result of an "inner" simulation of 2000 copies of the system. The inner simulation is performed for 10 million steps: the first 0.2 million steps are used for healing and the linear fit is performed for the last 9.8 million of steps of the run. The values of the fast coarse variables are initialized at the values at the end at the end of the previous inner integrator run.

Results of 3 calculations using the Newton method are shown in Fig. 7. An estimate of the equilibrium solution obtained from an equilibrium run with 500 copies of the system is shown by the circle. The 3 coarse Newton simulations are as follows:

1. Simulation 1 (crosses): the iterations started relatively close to the solution; in the derivative estimation, the increment for derivative calculation is $\Delta x = 5x_{err}$.

2. Simulation 2 (squares): the iterations started at the same point at which the coarse Euler simulation started, see Fig. 6; $\Delta x = 5x_{err}$.

3. Simulation 3 (triangles): the iterations started at the same point as those of the second run but now the derivative is computed with $\Delta x = x_{err}$.

The first Simulation converges to a point located close to the estimation of the equilibrium solution. The slight difference between the equilibrium estimate and the coarse Newton method result is probably due to fact that 500 copies (used in the control run) of the system do not provide as good statistics as 2000 copies (used in coarse Newton) do.

However, the situation is worse for the Simulations 2 and 3 started relatively far from the stationary point. These two calculations were terminated after the third iteration because it did not appear that the iterations were converging to a stationary solution. Moreover, these iterations are very sensitive to the choice of Δx: even the direction of Newton step is altered by changing Δx by a factor of 5. One reason for such a poor performance of the

Newton method far away from the stationary point is the high level of noise. In section VII we show that the function $\mathbf{F}(\mathbf{x})$ is very noisy if one only uses 2000 copies of the system. It is well known, from the simple deterministic context, that Newton iteration is quite sensitive (in a problem-dependent way) to the choice of initial conditions. It is mostly in a continuation context (i.e. when we have the solution at one parameter value, and want to find it at nearby parameter values) that Newton is routinely used. In the context of the present work, we only want to demonstrate that such continuum-inspired algorithms can work when applied appropriately and with sufficient variance reduction. Many deterministic variants of the Newton (including matrix-free Newton-Krylov methods) can be easily modified to work in a coarse timestepper context; which of these will be the least sensitive to noise, and what savings they can produce is the subject of current research. Once more, what we want to show here is that such continuum numerical analysis methods are, indeed, applicable as "wrappers" around the type of atomistic micellar simulations we perform. It is also important to notice that algorithms like Newton are capable of converging to unstable stationary states (e.g. transition states) and can be augmented to converge on *marginally stable* states (at the onset of instabilities)[2,3,5,31].

In addition to computing an equilibrium configuration of the system, results of the Newton method can be used to obtain the timescale (in MC iteration time) of approach of the coarse variables to this equilibrium. Eigenvalues of the coarse Jacobian near a stationary point correspond to timescales of the coarse variables. For the density variables, ν_m and σ_m^2, the eigenvalues computed near the equilibrium solution are on the order of 10^{-8}. This is in contrast with the eigenvalue 5×10^{-6} obtained in the companion paper for the evolution of a single micelle near its equilibrium configuration. These calculations confirm that there is at least a two order of magnitude separation of times cales between the coarse-grained dynamics of micelle density and the micelle size variables.

VII. KINETIC MONTE CARLO MODEL

In this section, we show that the poor performance of the Newton method is due to the high level of noise in the system. The simplest way to estimate the noise level in the evaluations of function $\mathbf{F}(\mathbf{x})$ is to compute values of this function on a relatively fine mesh of values of $\mathbf{x}$. Unfortunately, this task is formidable for our micellar system – a calculation of the function $\mathbf{F}(\mathbf{x})$ at a single point takes about 28 hours on a single AMD Athlon processor and, even with parallelization, the calculation on a fine mesh would take very long time.

In order to compute $\mathbf{F}(\mathbf{x})$ on a mesh and to prepare the ground for future "experiments" with various variance-reducing finite-difference methods, we consider the following simple kinetic Monte Carlo (KMC) model. We assume that there is a discrete set of possible states of the system and we number these states $0, 1, \ldots, N$. In the case of a micellar system, the i-th state corresponds to the system with i micelles in it. We consider N_{repeat} copies of the system and store the data in a form of a histogram $\mathbf{h} = (h_0, h_1, \ldots, h_N)$ so that the value of h_i is the number of copies of the system which are in the i-th state and $\sum_{i=0}^{N} h_i = N_{\text{repeat}}$. We further assume that the birth-death process is described by a first-order master equation,

$$\frac{d\mathbf{h}}{dt} = \mathbf{K}\mathbf{h}, \tag{16}$$

where $\mathbf{K}$ is the matrix of transition rates and k_{ij} is the rate of transition from state i to state j. It is shown in the companion paper that the birth-death of micelles can indeed be approximated by the first order kinetic process.

Although the solution of equation (16) is straightforward, in some cases this equation is not available in an explicit form. This happens, in particular, when it is not easy to identify the states of the system or the transition paths between these states. Examples of such systems include diffusion in random media and birth and death of micelles in a system with long-range interactions. In these cases, one has to resort to some microscopic integration tool such as MD or MC instead of solving the more macroscopic master equation.

In order to model the birth-death process and its interaction with the coarse integration tools, we use stochastic (kinetic Monte Carlo, KMC) simulation[32,33,34] to simulate the equation (16). In our simulations, we choose KMC parameters which model micelle birth/death process. The transition rates (i.e. rates of micelle birth and death) are chosen so that they satisfy the detailed balance condition consistent with the equilibrium distribution $P_{\text{eq}}(i)$, i.e.

$$\frac{P_{\text{eq}}(i-1)}{P_{\text{eq}}(i)} = \frac{k_{i,i-1}}{k_{i-1,i}}. \tag{17}$$

The equilibrium distribution is obtained from the full scale MC simulation.

The condition (17) guarantees convergence of the system to the equilibrium distribution. The rates of the system are chosen so that one unit of time of the KMC simulation corresponds to one million of GCMC steps. An example of a KMC simulation with thus chosen transition rates is shown in Fig. 8. Fig. 8a shows evolution of the probabilities $P(i) = h(i)/N_{\text{repeat}}$ and it is seen that these probabilities do approach an equilibrium. Evolution of the two moments (mean ν_m and standard deviation σ_m) of this distribution is shown in Fig. 8b and 8c. Comparison of Fig. 6 and Fig. 8 shows that the KMC model reproduces essential features of the evolution and hence its results can be used to analyse the accuracy coarse integration technique. Thus, in effect, in this model we have completely eliminated the fast dynamics and we focus on the rate-limiting slow dynamics – birth and death of micelles. We emphasize that "time" and "dynamics" here (as everywhere else in the paper) refer to the artificial GCMC dynamics.

Probably the most important feature of equation-free computation is that the algorithms are in effect "wrappers" which, through lifting and restriction subroutines, can be combined with *any* microscopic/atomistic simulator. In the same way we wrapped coarse projective integration and coarse Newton around the GCMC with our 7 coarse observables, we will now wrap it around the model kMC simulation (templated on the GCMC). We perform coarse projective Euler integration for the KMC model with 3 different numbers N_{repeat} of copies of the system. The observables here are the average and the standard deviation of the average ν_m and the standard deviation σ_m of the number of micelles, just as in the coarse integration of the full GCMC model. Results of the coarse kMC integration are shown in Figs. 8b and c by circles. The timestep of the Euler method is $\Delta t = 50$ and, before each projective step, the inner integration is performed for 10 units of time. The simulations with larger number of system copies (5×10^3 and 5×10^4) converge to the equilibrium solution. However, the simulation with 2×10^3 system copies exhibits strong oscillations around the stationary solution.

As evident from Figs. 8b and c, the accuracy of the coarse Euler method significantly depends on the number N_{repeat} of the system copies used in the microscopic simulation. In addition to N_{repeat}, the parameters of the coarse Euler method that can significantly affect the accuracy are the the length L of the microscopic simulation and a length α of the projection interval. For convenience, here we normalize α so that $\alpha = 1$ corresponds to projection over the internal microscopic simulator step Δt.

In order to investigate the accuracy of the coarse Euler projective step, we obtain evolution of coarse variables from the KMC simulation and perform the extrapolation of the coarse variables to various (normalized) timesteps α. We perform this projection for various values of realizations of the system N_{repeat} and for different lengths of the fitting interval L. For each set of parameters $(\alpha, N_{\text{repeat}}, L)$, we compute 10^4 realization of the "average trajectory", i.e. a trajectory which one obtains using N_{repeat} copies of the KMC system. These average trajectories are still noisy and hence the projections obtained from different realizations of these trajectories will have some scatter. We measure this scatter by computing the variance of 10^4 realizations of such projections. This variance, which we denote $\sigma^2(\alpha, N_{\text{repeat}}, L)$, measures the accuracy of the Euler step.

The value of N_{repeat} was varied from 10^2 to 10^4, α was varied from 10 to 10^3, and L was varied from 4 to 98. Initial conditions for the KMC simulation were taken to coincide with those of the simulations shown in Fig. 8. Here, we report results of the projection of ν_m. The conclusions for the other coarse variable, σ_m, are similar. Dependence of the variance of ν_m, $\sigma_\nu^2(\alpha, N_{\text{repeat}}, L)$, on the normalized projection time α is shown in Fig. 9a for $N_{\text{repeat}} = 2000$ and $L = 98$. The quadratic dependence of σ_ν^2 on α seen in Fig. 9a is also observed for all values of N_{repeat} and L considered in our numerical experiments. We conclude therefore that

$$\sigma^2(\alpha, N_{\text{repeat}}, L) = C(N_{\text{repeat}}, L)\alpha^2, \tag{18}$$

where $C(N_{\text{repeat}}, L)$ is the proportionality coefficient. We plot $C(N_{\text{repeat}}, L)$ for fixed $L = 98$ and various N_{repeat} as well as for fixed $N_{\text{repeat}} = 2000$ and various L in Fig. 9b and observe that, to a good accuracy, C is inversely proportional to both N_{repeat} and L. We therefore conclude that

$$\sigma_\nu^2(\alpha, N_{\text{repeat}}, L) \approx \frac{\alpha^2}{L N_{\text{repeat}}}. \tag{19}$$

Similar result is obtained for the accuracy of the projection of σ_m. The significance of the scaling Eq (19) is that there is a trade-off between the projection interval length α and the number of copies N_{repeat} and the interval length L used in the microscopic simulations. In other words, if we increase the projection length α, then we also need to increase the product NL by the same factor in order to keep variance of the projected data at the same level as the variance of the original data.

The efficiency of coarse projective integration, the choice of filtering, variance reduction[35] and extrapolation techniques it can be combined with, is the subject of current study; our group as well as other groups are comparing the efficiency of direct simulation to that of coarse integration methods.

Coarse Newton method for the kMC Model. We perform the coarse Newton computations for $N_{\text{repeat}} = 2 \times 10^3$ (the number of copies used in the coarse Newton method for micelles) and 5×10^4 (some large number that is unrealistic for the GCMC simulations). We perform the simulations with forward and central differences. In the case of the central differences, the Jacobian is obtained from the solution of the Eq. (14) in the least squares sense. The increment Δx for calculation of the Jacobian is chosen as follows: for simulations with $N_{\text{repeat}} = 2 \times 10^3$, we set $\Delta x = 5x_{err}$, where x_{err} is the error estimate of a coarse variable x. This choice is similar to that used in the implementation of the Newton method for GCMC simulations. However, such a choice is not practical for simulations with $N_{\text{repeat}} = 5 \times 10^4$, since in this case, the noise level is relatively low which leads to very small values of $\Delta x = 5x_{err}$. Hence, in the simulations with $N_{\text{repeat}} = 5 \times 10^4$, we use increment $\Delta x = 0.05$. This value roughly corresponds to the increments used in the simulations with $N_{\text{repeat}} = 2 \times 10^3$.

Iterations of the coarse Newton method are shown in Figs. 10 and 11. The simulations shown in Fig. 10, are started from a point located relatively close to the equilibrium and the simulations shown in Fig. 11 are started from a point located further away from the equilibrium. The iterations with $N_{\text{repeat}} = 5 \times 10^4$ converge relatively fast to the equilibrium and the convergence is significantly faster if one uses the central difference algorithm. However, for $N_{\text{repeat}} = 2000$ (the number of copies which is realistic in our GCMC simulations of the micellar system), the rate of convergence is slower and, moreover, the central differences are not guaranteed to provide an improvement of convergence. In fact, in the simulations shown in Fig. 10, the central differences perform significantly worse than the forward differences. Reliable estimation for the (coarse) derivative of a noisy function is a vital element in the bridging of microscopic similations with continuum-type numerical algorithms based on (macroscopic) smoothness and Taylor series.

Finally, in order to estimate the level of noise, we compute one of the nonlinear functions (F_1 = slope of ν_m) for the Newton method and plot it in Fig. 12 for $N_{\text{repeat}} = 2 \times 10^3$, $N_{\text{repeat}} = 5 \times 10^4$ and an analytic result obtained from the solution of the master equation. We observe that the right-hand side is very noisy even for $N_{\text{repeat}} = 5 \times 10^4$ and is extremely noisy for $N_{\text{repeat}} = 2 \times 10^3$. This explains such a poor performance of the coarse Newton method for $N_{\text{repeat}} = 2 \times 10^3$ and suggest that a significant effort should be directed towards developing variance-reducing schemes for the Newton method[30,36].

VIII. DISCUSSION

We have demonstrated the application of coarse-grained, equation-free computational techniques to the GCMC simulations of the micellar system (and to their kMC caricature). Of our 7 coarse variables, 5 "fast" ones (small cluster parameters λ, ν_s, and σ_s and the micelle size variables ν and σ) are observed to be slaved to two slow (master) variables: average ν_m and the standard deviation σ_m of the number of micelles in the system.

Although the current coarse computation results do not show significant improvement of the efficiency of the coarse integration as compared to the full-scale MC simulations, several possible improvements of the coarse integration methods are currently a subject of active research.

Although in our micellar system, we have already taken advantage of the separation of time scales between the 5 fast coarse variables and the 2 slow coarse variables, the dynamics of the 2 slow coarse variables (ν_m and σ_m) has not been fully explored. In particular, it is possible to improve the efficiency of the coarse computation by a different choice of the slow coarse variables that describe the distribution $P(N_{\text{micelle}})$ of the number N_{micelle} of micelles in a simulation box. Recall that in the current work, these variables are chosen to be the mean ν_m and the standard deviation σ_m of N_{micelle} and N_{micelle} is assumed to have a truncated Gaussian distribution. A more detailed analysis of the equation (16), which models evolution $P(N_{\text{micelle}})$, shows a separation of timescales which is absent in the dynamics of ν_m and σ_m. This may suggest different coarse variables to represent $P(N_{\text{micelle}})$, that may better reflect the timescale separation in the evolution of $P(N_{\text{micelle}})$.

Another possibility for improvement of the accuracy of the coarse Euler method lies in the improvement of the projective step. In the current paper, we have used a linear least squares fit to extrapolate the values of the coarse variables. We assumed here that the evolution of the coarse variables can be described by a deterministic equation. These deterministic coarse variables are ensemble averages of stochastic variables (such as number of micelles in a simulation box, considered in the current papers). It may be advantageous to take the stochasticity of this dynamics into account in the projective integration step; instead of fitting a deterministic model one may attempt to fit a *stochastic* model for the evolution of the observables, as is done in the companion paper[22]. Better extrapolation schemes (e.g. based on maximum likelihood estimation and templated on Adams-Bashforth methods[24]) which would incorporate filtering and take into account the correlations between values of the coarse variables at consecutive time steps, may lead to more efficient algorithms. Comparable statements also apply to the coarse Newton method.

The point of this paper has been the illustration of a possible bridge between traditional, continuum numerical methods and modern atomistic/microscopic simulations (here, GCMC simulations of micelle formation) Given the appropriate coarse observables, the detailed (here GCMC) simulator is initialized *conditioned on* the observables and then evolved for a (macroscopically) short time. The computational data are used to fit a local macroscopic evolution equation which we assume exists. In this paper we fit only the deterministic component of this equation; more generally one can try to fit a local stochastic differential equation. The local model is then used to design appropriate initial conditions for new, subsequent computational experiments with the detailed simulator; the protocol for this "design of computational experiments" is provided by continuum numerical algorithms, such as initial value problem solvers (e.g. forward Euler), fixed point solvers (e.g. Newton Raphson) eigensolvers etc. Smoothness (Taylor series) at the level of the macroscopic observables is the underlying point of these methods; and what makes them possible is the ability to initialize the microscopic simulator essentially "at will", consistent with macroscopic observables.

It is appropriate to close this paper with a short "advertisement" for the companion one[22]. In this paper we assumed an underlying smooth deterministic model for the expected behavior of the macroscopic observables. For certain stochastic systems (exemplified by a particle in a double well under the effect of noise), the long term behavior of the system statistics (approach to a final equilibrium density) may well be modeled by continuum equations. One has to collect simulation data for a single particle over long times, before one can observe the rate of density evolution. For such systems, it may be more appropriate to fit the short term dynamics in terms of a stochastic differential equation (e.g. a Langevin-type equation). The long characteristic times for the equilibration of our simulation are indeed governed by rare events (the formation and destruction of micelles). In the companion paper we will show how to use similar "coarse computation" methods to design experiments based on an *effective stochastic evolution equation* for the macroscopic observables, rather than the *effective deterministic evolution equation* we used here.

Acknowledgements. This work was partially supported by AFOSR and an NSF/ITR grant. It is a pleasure to acknowledge discussions with Prof. C. W. Gear and Dr. G.

Hummer.

[*] Current address: Department of Chemical Engineering University of Florida, Gainesville, FL 32611; Electronic address: dkopelevich@che.ufl.edu

[†] Electronic address: azp@princeton.edu

[‡] Corresponding author; Electronic address: yannis@arnold.princeton.edu

[1] C. Theodoropoulos, Y.-H. Qian, and I. G. Kevrekidis, Proc. Natl. Acad. Sci. **97**, 9840 (2000).

[2] C. I. Siettos, M. D. Graham, and I. G. Kevrekidis, J. Chem. Phys. **118**, 10149 (2003), can be obtained as cond-mat/0211455 at arXiv.org.

[3] G. Hummer and I. G. Kevrekidis, J. Chem. Phys. **118**, 10762 (2003), can be obtained as physics/0212108 at arXiv.org.

[4] S. L. J. Cisternas, C. W. Gear and I. G. Kevrekidis, submitted to Proc. Roy. Soc. London (2003), can be found as nlin.AO/0310011 at arXiv.org.

[5] A. G. Makeev, D. Maroudas, and I. G. Kevrekidis, J. Chem. Phys. **116**, 10083 (2002).

[6] A. G. Makeev, D. Maroudas, A. Z. Panagiotopoulos, and I. G. Kevrekidis, J. Chem. Phys. **117**, 8229 (2002).

[7] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg, and K. Theodoropoulos, Comm. Math. Sciences **1**, 715 (2003), original version can be obtained as physics/0209043 at arXiv.org.

[8] I. G. Kevrekidis, C. W. Gear, and G. Hummer, AIChE Journal **50**, 1346 (2004).

[9] C. W. Gear and I. G. Kevrekidis, SIAM J. Sci. Comput. **24**, 1091 (2003), also NEC Technical Report NECI-TR 2001-029, can be obtained as http://www.neci.nj.nec.com/homepages/cwg/projective.pdf.

[10] J. Israelachvili, *Intermolecular and Surface Forces* (Wiley, New York, 1989), 2nd ed.

[11] R. G. Larson, L. E. Scriven, and H. T. Davis, J. Chem. Phys. **83**, 2411 (1985).

[12] R. G. Larson, Journal de physique II **6**, 1441 (1996).

[13] M. A. Floriano, E. Caponetti, and A. Z. Panagiotopoulos, Langmuir **15**, 3143 (1999).

[14] A. Z. Panagiotopoulos, M. A. Floriano, and S. K. Kumar, Langmuir **18**, 2940 (2002).

[15] A. M. Ferrenberg and R. H. Swendsen, Phys. Rev. Lett. **61**, 2635 (1988).

[16] A. M. Ferrenberg and R. H. Swendsen, Phys. Rev. Lett. **63**, 1195 (1989).

[17] I. T. Jolliffe, *Principal Component Analysis* (Springer-Verlag, New York, NY, 1986).

[18] A. J. Smola, O. L. Mangasarian, and B. Schoelkopf, in *24th Annual Conference of Gesselschaft für Klassifikation* (University of Passau, Passau, Germany, 2000), data Mining Institute Technical Reort 99-04.

[19] J. P. Ryckaert, G. Ciccotti, and H. Berendsen, J. Comp. Phys. **23**, 327 (1977).

[20] C. W. Gear, I. G. Kevrekidis, and C. Theodoropoulos, Comput. Chem. Eng. **26**, 941 (2002).

[21] C. W. Gear (2001), technical Report NEC TR 2001-130.

[22] D. I. Kopelevich, A. Z. Panagiotopoulos, and I. G. Kevrekidis, Submitted to J. Chem. Phys. (2004).

[23] G. M. Torrie and J. P. Valleau, Chem. Phys. Letters **28**, 578 (1974).

[24] R. Rico-Martínez, C. W. Gear, and I. G. Kevrekidis, J. Comp. Phys. **196**, 474 (2004).

[25] D. T. Gillespie, J. Comp. Phys. **22**, 403 (1976).

[26] D. T. Gillespie, J. Phys. Chem. **81**, 2340 (1977).

[27] C. W. Gear (2002), technical Report NECI TR 2002-025N, can be obtained as http://www.neci.nj.nec.com/homepages/cwg/steadystate.pdf.

[28] T. O. Drews, R. D. Braatz, and R. C. Alkire, J. Electrochem. Soc. **150**, C807 (2003).

[29] C. T. Kelley, *Iterative Methods for Optimization* (SIAM, Philadelphia, 1999).

[30] J. C. Spall, IEEE Trans. Automat. Contr. **45**, 1839 (2000).

[31] C. I. Siettos, I. G. Kevrekidis, and D. Maroudas, Int. J. Bifurcations and chaos **14**, 207 (2004).

[32] J. S. Shaffer and A. K. Chakraborty, Macromolecules **26**, 1120 (1993).

[33] H. C. Kang and W. H. Weinberg, J. Chem. Phys. **90**, 2824 (1989).

[34] R. Q. Snurr, A. T. Bell, and D. N. Theodorou, J. Phys. Chem. **98**, 11948 (1994).

[35] H. C. Öttinger, *Stochastic Processes in Polymeric Fluids. Tools and Examples for Developing Simulation Algorithms* (Sringer-Verlag, Berlin-Heidelberg, 1996).

[36] M. Melchior and H. C. Öttinger, J. Chem. Phys **103**, 9506 (1995).

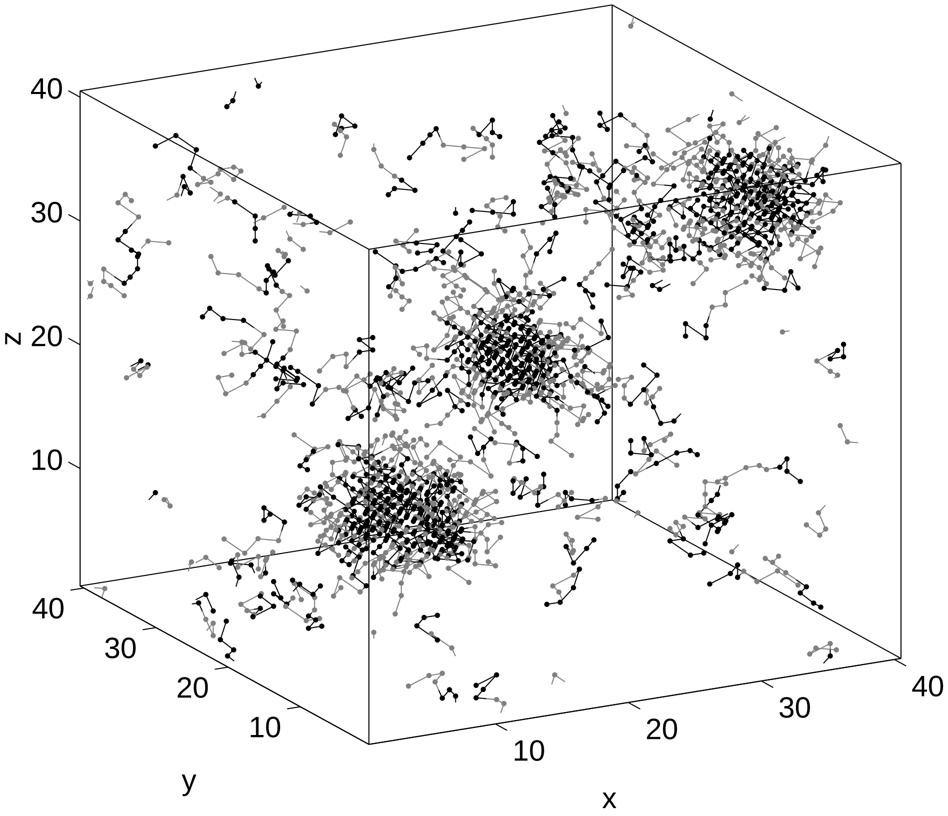

FIG. 1: Snapshot of a micellar system for temperature $T = 7.0$ and chemical potential $\mu = -47.40$. Hydrophobic tail beads are shown in black and hydrophilic head beads are shown in gray

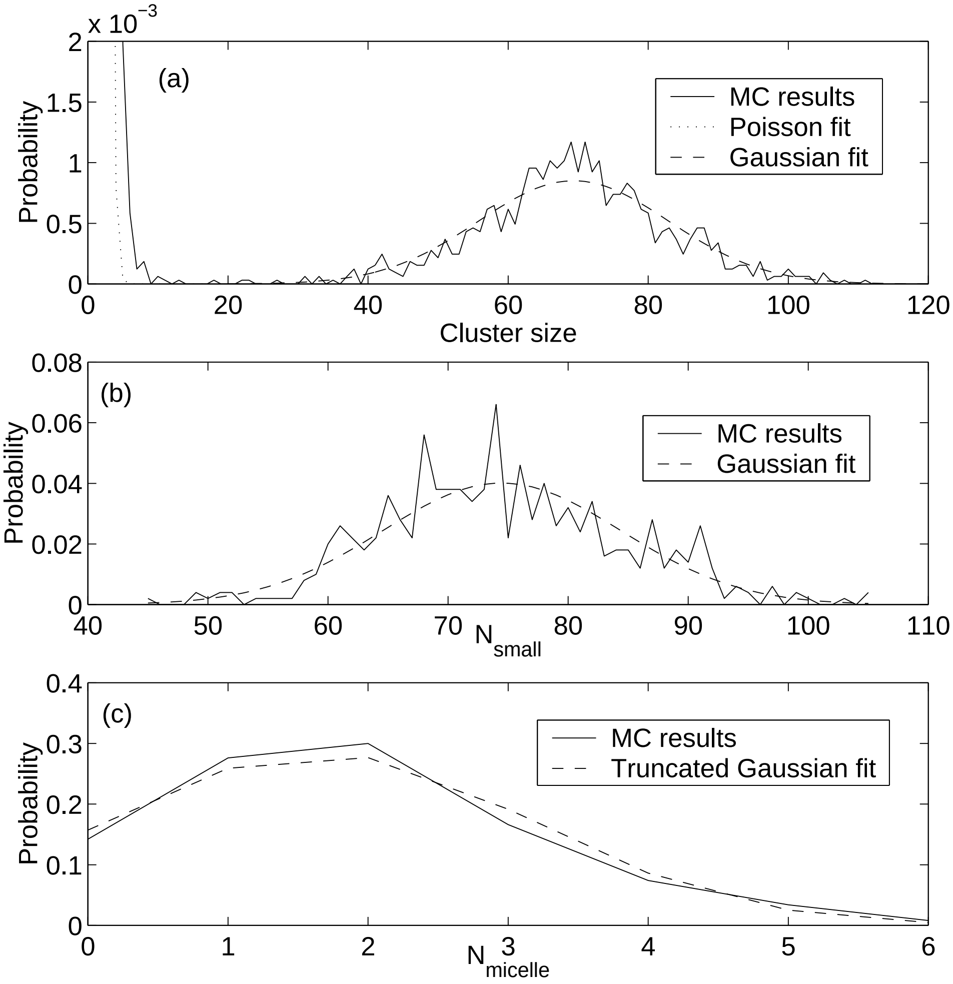

FIG. 2: Definition of coarse variables: (a) cluster size distribution, (b) distribution of number of molecules N_{small} contained in small clusters, and (c) distribution of number N_{micelle} of micelles. Distributions shown in this plot are obtained from averaging of 500 MC realizations.

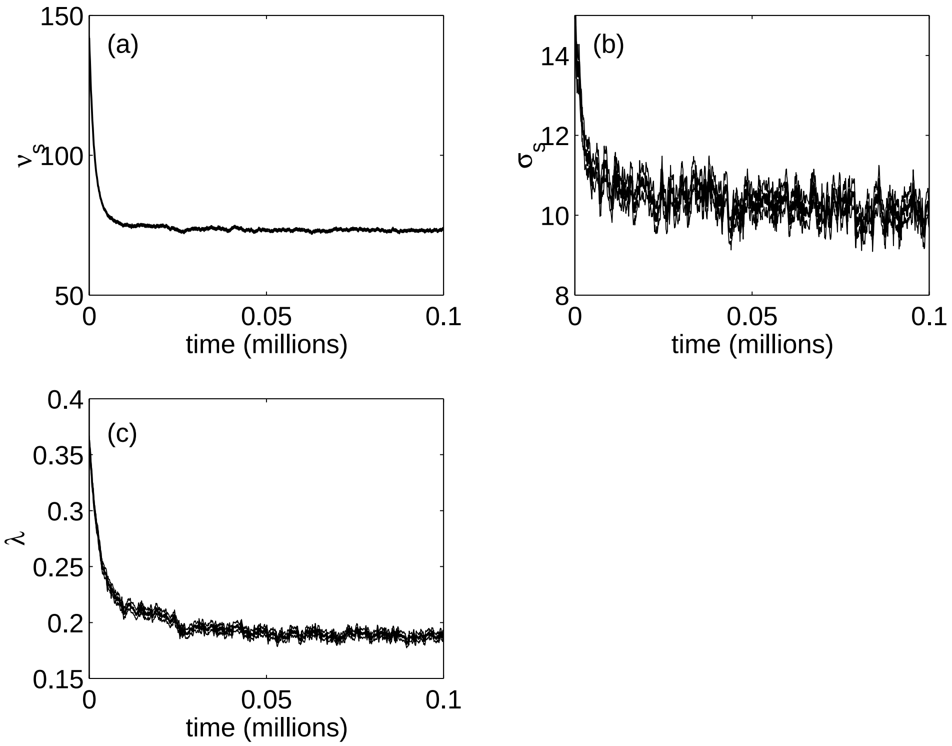

FIG. 3: Evolution of the small cluster parameters as the temperature and the chemical potential are switched from $T = 7.5$, $\mu = -46.20$ to $T = 7.0$, $\mu = -47.40$: (a) average small cluster size ν_s, (b) standard deviation σ_s of the small cluster size, and (c) average number λ of molecules contained in the small clusters The plots are obtained from averaging of 500 MC realizations. The error bars are indicated by the thin lines.

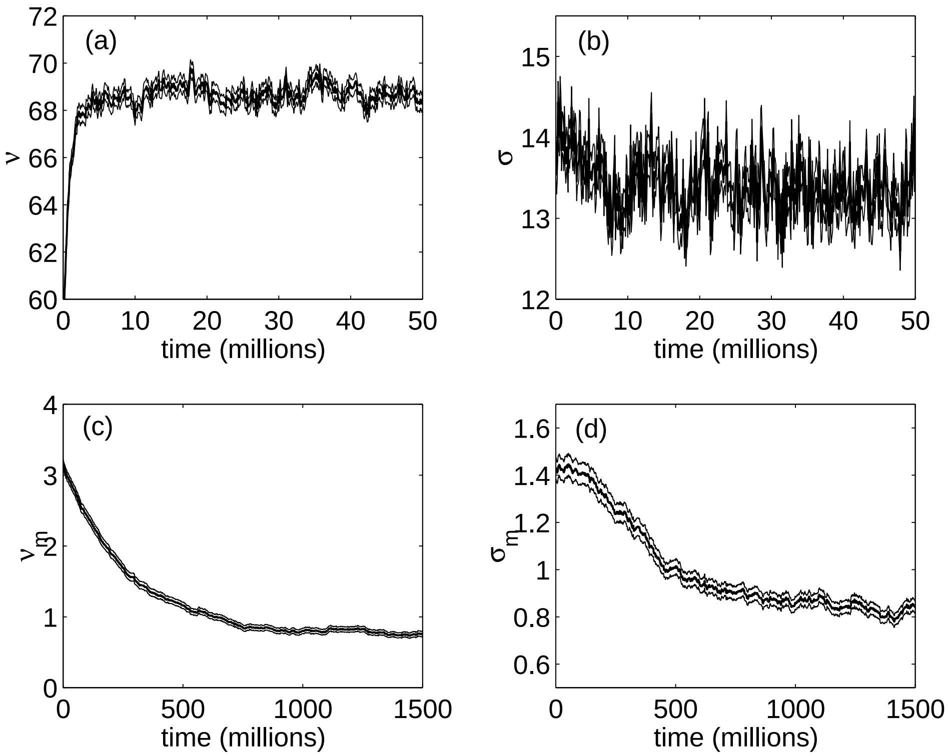

FIG. 4: Evolution of the micelle parameters as the temperature and the chemical potential are switched from $T = 7.5$, $\mu = -46.20$ to $T = 7.0$, $\mu = -47.40$: (a) average micelle size ν, (b) standard deviation σ of the micelle size, (c) average number ν_m of micelles in the system, and (d) standard deviation σ_m of number of micelles in the system. The plots are obtained from averaging of 500 MC realizations. The error bars are indicated by the thin lines.

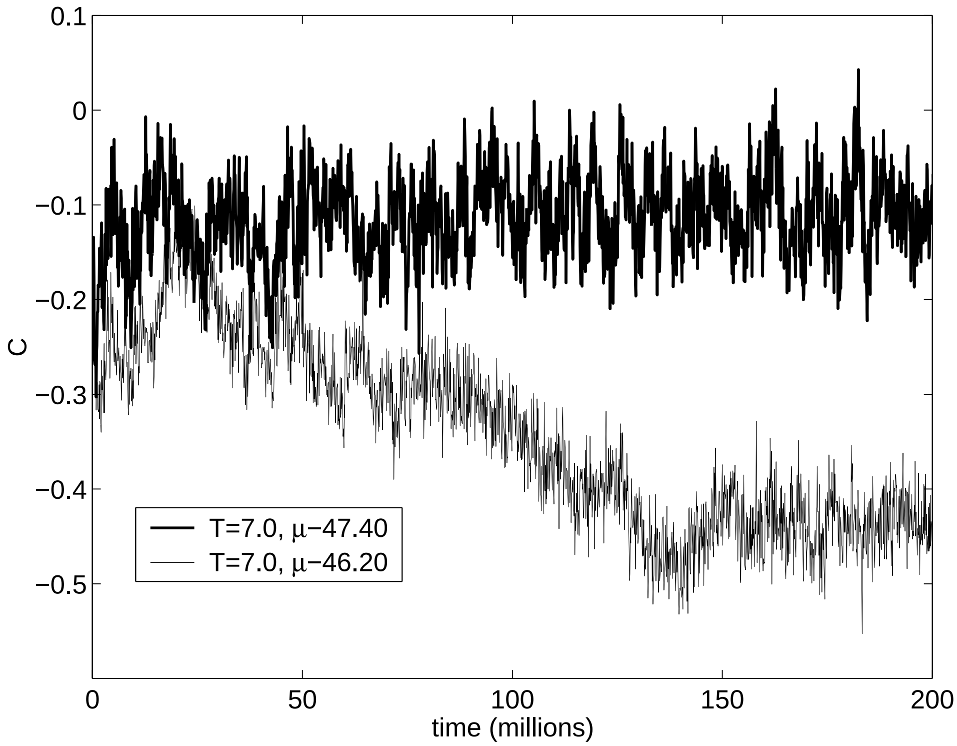

FIG. 5: Correlation coefficient between N_{micelle} and the average micelle size. Shown are simulation results for two sets of parameter values: low surfactant density, $k_BT = 7.0, \mu = -47.40$ (thick line) and high surfactant density, $k_BT = 7.0, \mu = -46.20$ (thin line). Both of these simulations have been initialized from a configuration at $k_BT = 7.5, \mu = -46.20$. The reported correlation functions are obtained from averaging of 500 MC realizations.

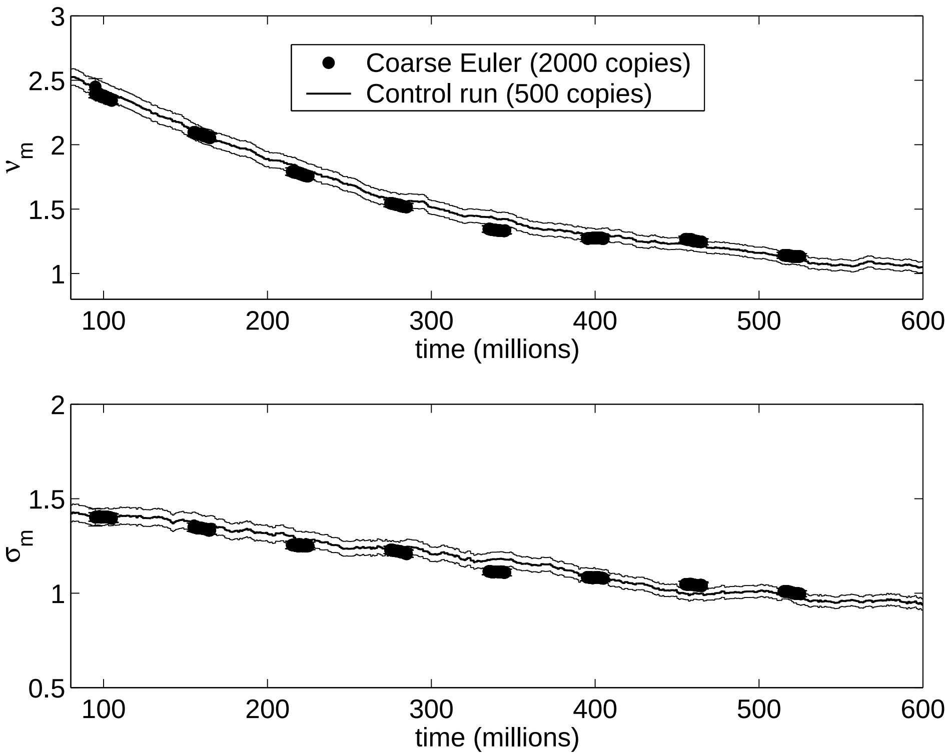

FIG. 6: Results of coarse Euler method: the solid lines correspond to a control run (thin lines show the error bars), and the circles show the evolution of projected coarse variables; $t_{\mathrm{healing}} = 0.2$ million, $t_{\mathrm{generation}} = 9.8$ million, $\Delta t = 50$ million. The error bars for the coarse integration are relatively small and are almost invisible in the plot.

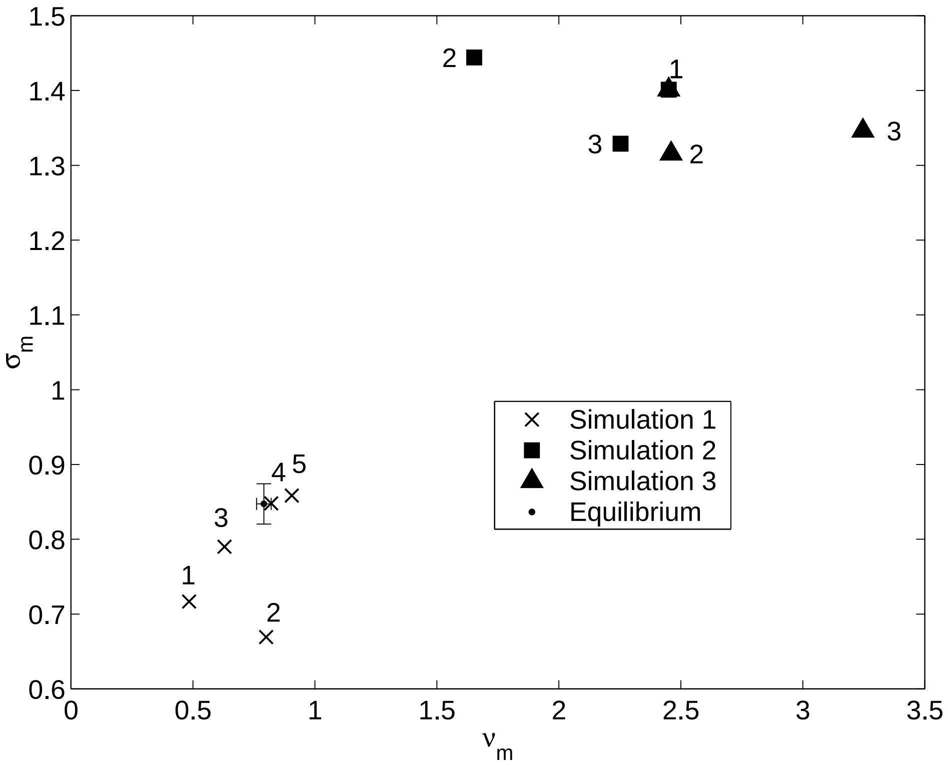

FIG. 7: Results of the coarse Newton method. The numbers on the plot indicate the iteration number. See text for details.

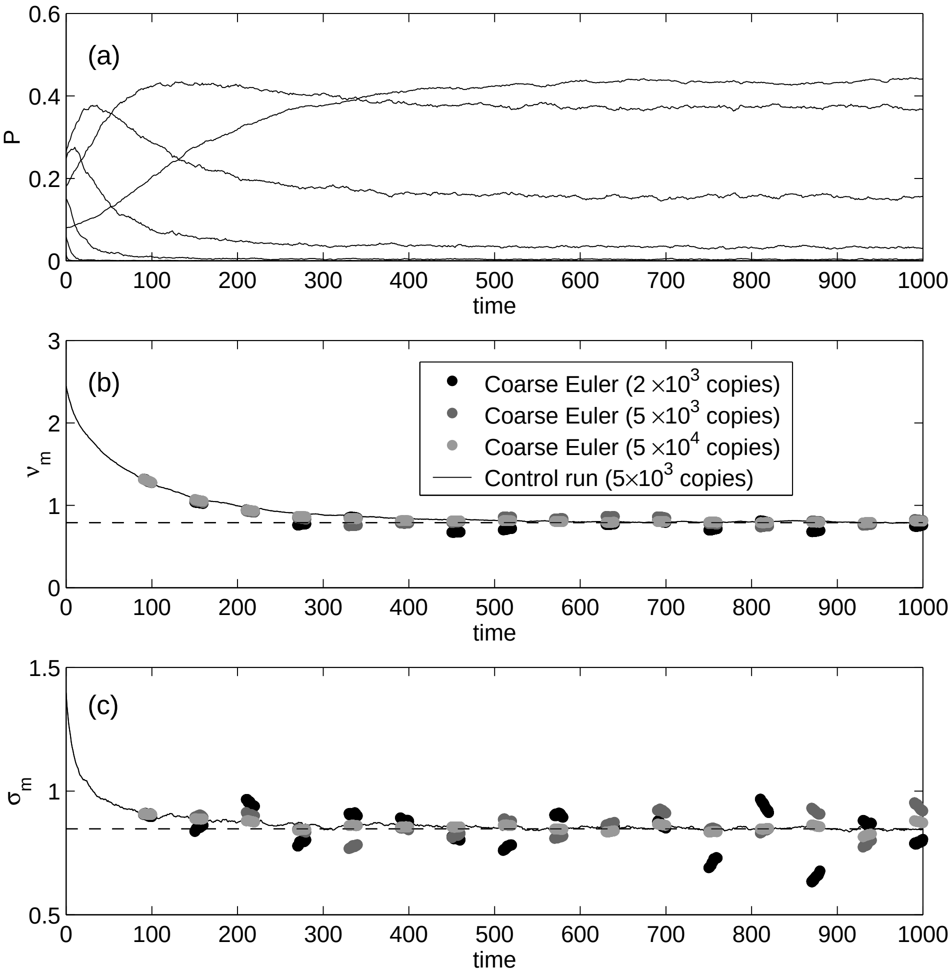

FIG. 8: Results of KMC simulations with $N_{\text{repeat}} = 5000$ copies of the system (solid lines): (a) evolution of probability distribution $P(i)$, each line corresponds to a different state i, $i = 0, \ldots, 7$; (b) evolution of the mean ν_m, and (c) evolution of the standard deviation σ_m. Time here is measured in millions of GCMC steps. Circles in plots (b) and (c) show results of the coarse projective Euler method for $N_{\text{repeat}} = 2 \times 10^3$ copies (black circles), $N_{\text{repeat}} = 5 \times 10^3$ copies (dark gray circles), and $N_{\text{repeat}} = 5 \times 10^4$ copies (light gray circles).

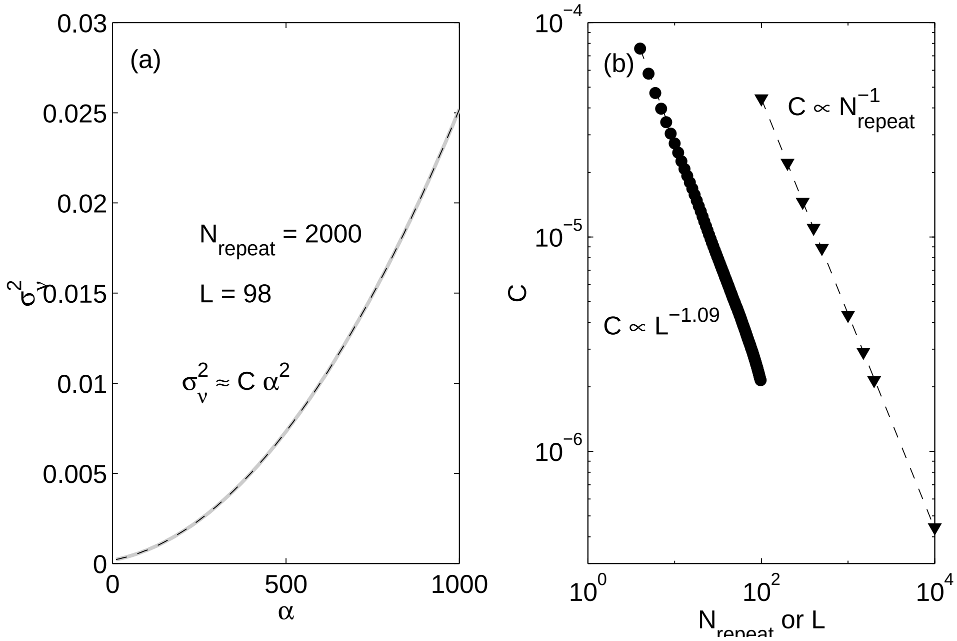

FIG. 9: (a) Estimated variance $\sigma_\nu^2(\alpha)$ of projection for $N_{\mathrm{repeat}} = 2000$, $L = 98$ as a function of the normalized extrapolation time α; the thick gray line shows the KMC results and the thin black dashed line shows the quadratic fit; (b) triangles: scaling of the coefficient C with respect to N_{repeat}, $L = 98$ is fixed; circles: scaling of C with respect to L; $N_{\mathrm{repeat}} = 2000$ is fixed. The dashed lines show the logarithmic fit.

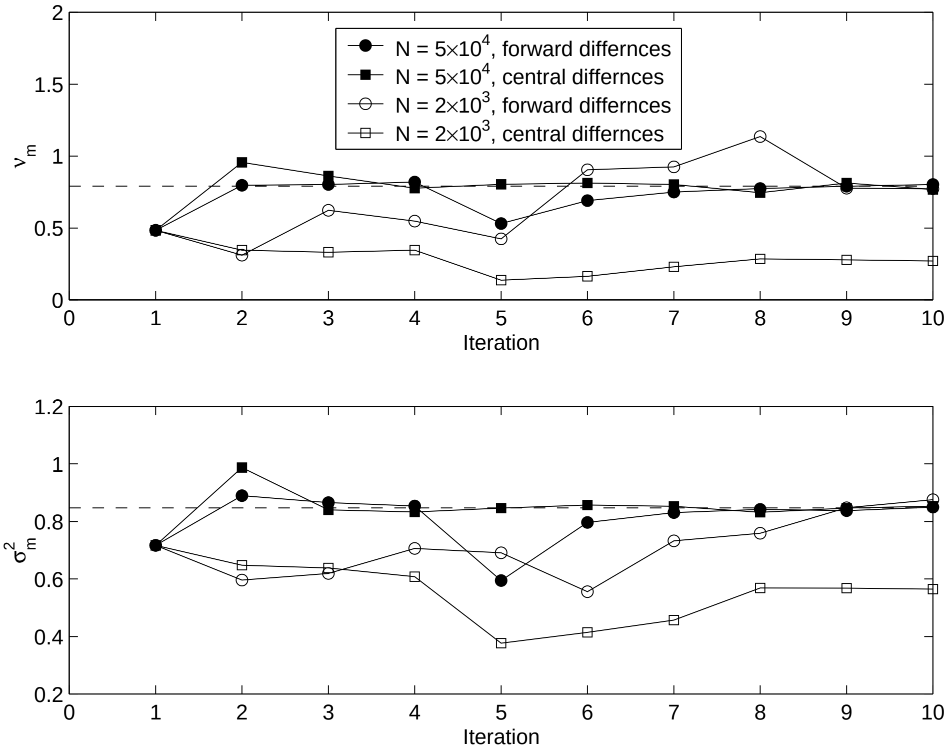

FIG. 10: Iterations of the Newton method for the KMC model. Open symbols show simulations with $N_{\mathrm{repeat}} = 2 \times 10^3$ and closed symbols show simulations with $N_{\mathrm{repeat}} = 5 \times 10^4$. Circles show iterations of the Newton method with the forward difference estimate of the Jacobian and squares show the iterations with the central difference estimate of the Jacobian. Dashed lines show the estimate of the equilibrium. Solid lines are shown to guide the eye.

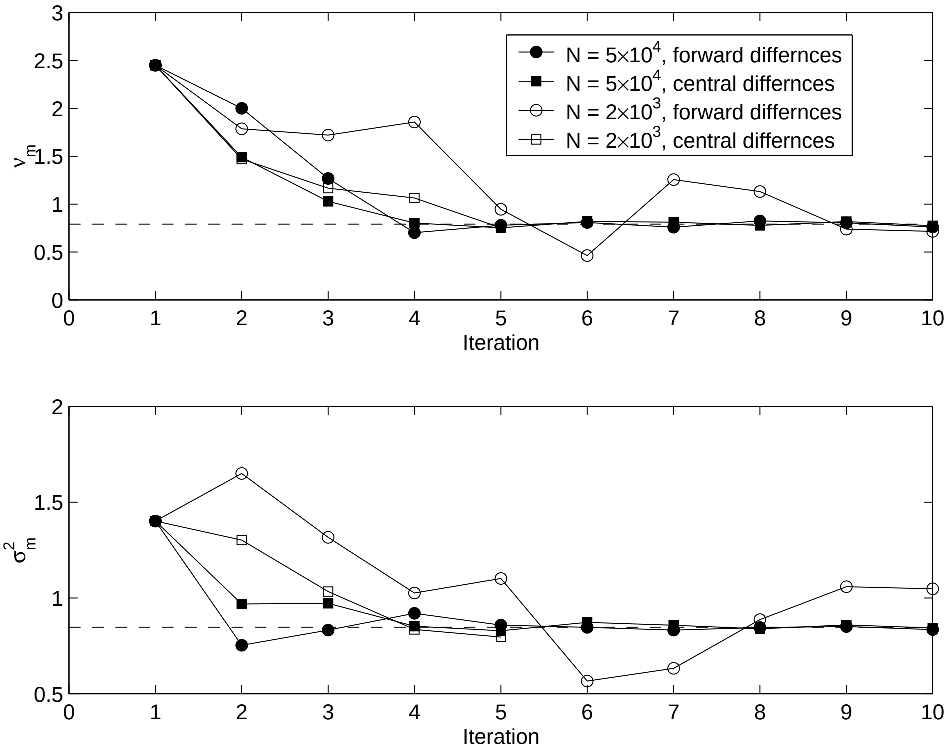

FIG. 11: Iterations of the Newton method for the KMC model. The same simulations as shown in Fig. 10 except here the initial point is chosen further away from the equilibrium solution.

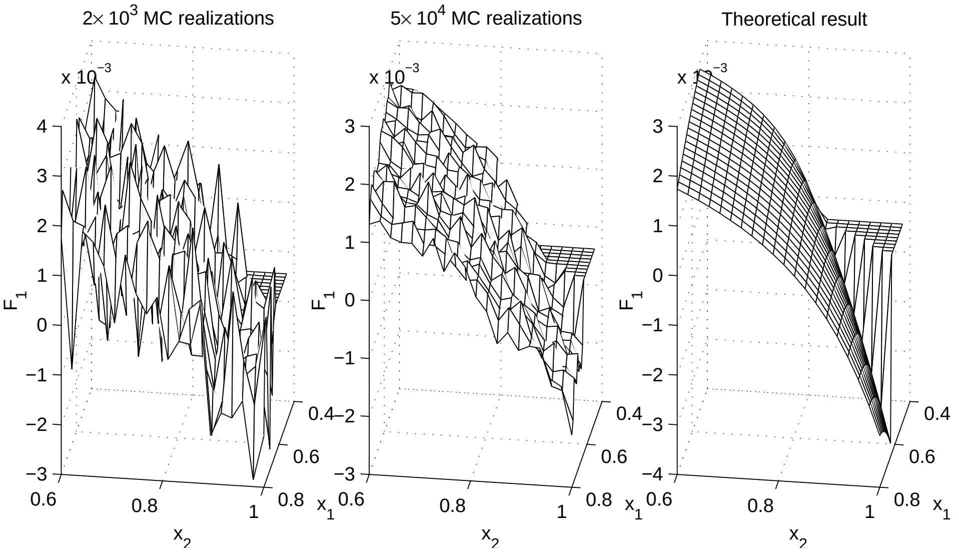

FIG. 12: F_1 obtained from KMC simulations and the theoretical prediction